UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/616,293	07/08/2003	Charles Simonyi	408068004US	1592

25096 7590 03/25/2005

PERKINS COIE LLP
PATENT-SEA
P.O. BOX 1247
SEATTLE, WA 98111-1247

EXAMINER	
THERIAULT, STEVEN B	

ART UNIT	PAPER NUMBER
2179	

DATE MAILED: 03/25/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/616,293	SIMONYI, CHARLES	
	Examiner	Art Unit	
	Steven B. Theriault	2179	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *08 July 2003*.

2a)□ This action is **FINAL**. 2b)☒ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-46* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-46* is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)□ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *08 July 2003* is/are: a)☒ accepted or b)□ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)□ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)□ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)□ All b)□ Some * c)□ None of:

1.□ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____.

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date *11/21/2003*.

4) □ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) □ Notice of Informal Patent Application (PTO-152)

6) □ Other: _____.

DETAILED ACTION

1. This action is responsive to the following communications: The original application filed on 07/08/2003 and an information disclosure statement filed 11/21/2003.

2. Claims 1-46 are pending in the case. Claims 1, 9, 15, 22, 28, 33, and 42 are the independent claims.

Applicant's attention is directed to the fact that a new examiner has been assigned to this case. The Examiner's name and telephone number are provided below.

Claim Rejections - 35 USC § 102

3. **The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:**

A person shall be entitled to a patent unless –

((e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

4. **Claims 15-21 are rejected under 35 U.S.C. 102(e) as being anticipated by Little et al (hereinafter Little) U.S. Patent Publication No. 2002/0091990 A1 issued Jul. 11, 2002.**

In regard to **Independent claim 15,** Little teaches the method for providing information for a computer program, the method comprising:

- *Providing a hierarchy of schemas defining valid computer programs, each schema specifying program element types at different levels of abstraction*; (Little figure 1-28) Little shows the methods, classes and attributes and shows other classes in the package which are shown in different levels of abstraction.

- *Providing a computer program with program elements, each program element having a program element type*; (Little figure 20 and page 10, column 2, lines 1-30) Little teaches a class diagram that represents a program and that contains nodes or icons graphics that are configured within a tree. Each node is an object or class with the respective inherited or dependent classes. Little shows the tree visually on the right and on the left in the non-visual tree component. The node type are the class type as specified in the diagram.
- *Identifying a derivation of program element types for a program element of the provided computer program from the provided hierarchy of schemas; and* (Little figure 1-28) Little shows the inherited classes and methods in the class diagrams.
- *Displaying an indication of program element types in the identified derivation*. (Little figure 1-28) Little shows the inherited classes and methods in the class diagrams.

With respect to **dependent claim 16,** Little teaches the [*method where the identifying of a derivation is performed in response to a user selecting a program element of the provided computer program*.] (Little figure 1-28) Little shows as the user selects the method or object or class where the element is located in the tree structure on the left of diagrams 17-20.

With respect to **dependent claim 17,** Little teaches the [*method including upon receiving a selection of a displayed program element type, displaying the information associated with the selected program element type*.] (Little figure 1-28) Little shows that if the user were to select a method all of the associated information would be displayed under the method. Little also shows the classes in the class diagram, which includes the associated information.

With respect to **dependent claim 18,** Little teaches the [*method where the displaying to the user of an indication of each program element type includes displaying name information associated with a program element type.*] (Little figure 1-28) Little shows the displaying of program types (objects and/or classes and methods) with naming information.

With respect to **dependent claim 19,** Little teaches the [*method of where the information associated with a program element type is stored as an attribute of the program element type.*] (Little figure 1-28) Little shows attributes within the class, which is of a specific type and would be stored or embedded within the class. An attribute is information within a class.

With respect to **dependent claim 20, Little** teaches the [*method where a program element type is stored as a definition within the computer program.*] (Little figure 1-28) Little shows a class and the elements of a program and the classes are stored as a class within the program and have keywords identifying them as classes.

With respect to **dependent claim 21,** Little teaches the [*method where a program element type that is a specific instance of a more abstract program element type is specified by an is a relationship.*] (Little figure 1-28) Little shows objects that inherit from a parent class, which are "is a" relationships.

References to specific columns, figures or lines should not be limiting in any way. The entire reference provides disclosure related to the claimed invention.

Claim Rejections - 35 USC § 103:

5. **The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:**

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. **Claims 1-14, 22-46 are rejected under 35 U.S.C. 103(a) as being unpatentable over Little et al (hereinafter Little) U.S. Patent Publication No. 2002/0091990 A1 issued Jul. 11, 2002, and filed Oct. 4, 2001, and in view of DeStefano et al (hereinafter DeStefano) U.S. Patent No. 6,275,227 B1 issued Aug. 14, 2001, and filed Feb. 9, 1998.**

In regard to **Independent claim 1,** Little teaches a *method in a computer system for providing help information for a computer program, the method comprising*:

- *Providing a program tree representation of the computer program, the program tree having program elements, each program element having a program element type*; (Little figure 20 and page 10, column 2, lines 1-30) Little teaches a class diagram that represents a program and that contains nodes or icons graphics that are configured within a tree. Each node is an object or class with the respective inherited or dependent classes. Little shows the tree visually on the right and on the left in the non-visual tree component. The node type is the class type as specified in the diagram.
- *Identifying ancestor program element types of the selected program element*; (Little figure 20) Little shows the hierarchy of the class structure in the tree node.
- *Displaying to the user an indication of each identified ancestor program element type*; (Little figure 1-28) Little gives an indication of the super class or master class in the diagram as showing which component inherits from the super class or implements the interfaces classes.

Little fails to expressly disclose:

- *Upon receiving a selection of a displayed ancestor program element type, displaying to the **user help information associated** with the selected ancestor program element type.*

- *Providing a hierarchy of program element types, each program element type having* ***associated help information****;*
- *Receiving from a user an indication* ***to provide help information for a selected program*** *element;*

DeStefano discloses a user interface control for displaying multiple levels of user information based on the amount of detail that the user needs to perform the task. Individual elements within the program, which can be component, parameter, method or other variables and attributes, are assigned a level identifier, which categorizes the information into levels of abstraction. DeStefano teaches an example combo box where the instructional material is provided on how to enter data into the combo box. The user is able to drive down further to get more detailed information on the range of input and then drive down another level to get the reasons for the ranges. (DeStefano column 8, lines 39-67 and column 9, lines 35-45). DeStefano and Little are analogous art because they are from the same field of endeavor of displaying user information about the graphical user interface.

Accordingly, It would have been obvious to one of ordinary skill in the art, having the teachings of Little and DeStefano before him at the time of the invention was made, to modify the system of Little to incorporate the multiple levels of help that can be assigned to any component or object as taught by DeStefano, in order to obtain a system that is able to element based help information. One would have been motivated to make such a combination because of the need to have an improved manner of integrating together the presentation of interface controls with instructional material as taught by DeStefano.

With respect to **dependent claim 2,** Little teaches the [*method where the displaying to the user of an indication of each ancestor program element type includes displaying naming information associated with a program element type*]. (Little figure 1-28) Little shows the displaying of ancestor program types with naming information.

With respect to **dependent claim 3,** claim 3 incorporates substantially similar subject matter as claimed in claim 1 and is rejected along the same rationale.

With respect to **dependent claim 4,** Little teaches [*the method where program element types are stored in schemas defining valid program trees at different levels of abstraction*] (Little figure 1-28) Little shows program trees in class diagrams that depict different levels of abstraction.

With respect to **dependent claim 5,** Little teaches [*the method where a schema is represented as a tree data structure*] (Little figure 1-28) Little shows program trees in class diagrams that depict data, as it is stored in a database.

With respect to **dependent claim 6,** Little teaches the [*method where each program element type is represented as a node within a tree.*] (Little figure 1-28) Little shows each iconic class identifier or object as a node within a tree. Little also shows each element in a tree node on the left in the diagram.

With respect to **dependent claim 7,** Little teaches the [*method where the ancestor program element types are specified by is a relationships, starting at the selected program element.*] (Little figure 1-28) Little shows objects that inherit from a parent class, which are "is a" relationships.

With respect to **dependent claim 8,** Little teaches the [*method where an ancestor program element type includes a program element type defined in the program tree.*] (Little figure 1-28) Little teaches an element type that is defined in the program tree.

In regard to **Independent claim 9,** Little teaches *the method for providing help information for a computer program, the method comprising:*

- *Providing a computer program with program elements, each program element having a program element type*; (Little figure 1-28) Little teaches the program elements that are a class, an interface, a method, a variable, or an attribute.
- *Identifying a derivation of program element types for the selected program element*; (Little figure 1-28) Little teaches which classes inherit or implement or are abstract classes by the common notation shown in the UML drawings. Little also shows the package structure in the package tree on the left of figures 17-20.
- Displaying to the user an indication of program element types in the identified derivation; (Little figure 1-28) Little teaches which classes inherit or implement or are abstract classes by the common notation shown in the UML drawings.

Little fails to expressly disclose:

- Upon receiving a selection of a displayed program element type, displaying to the **user help information associated with the selected program element type**.
- Receiving from a user an indication to **provide help information** for a selected program element;
- Providing a specification of program element types, a program element type being defined as a specific instance derived from a more general program element type, program element types **having associated help information**;

DeStefano discloses a user interface control for displaying multiple levels of user information based on the amount of detail that the user needs to perform the task. Individual elements within the program, which can be component, parameter, method or other variables and attributes, are assigned a level identifier, which categorizes the information into levels of abstraction. DeStefano teaches an example combo box where the instructional material is provided on how to enter data into the combo box. The user is able to drive down further to get more detailed information on the range of input and then drive down another level to get the reasons for the ranges. (DeStefano column 8, lines 39-67 and column 9, lines 35-45). DeStefano and Little are analogous art because they are from the same field of endeavor of displaying user information about the graphical user interface.

Accordingly, It would have been obvious to one of ordinary skill in the art, having the teachings of Little and DeStefano before him at the time of the invention was made, to modify the system of Little to incorporate the multiple levels of help that can be assigned to any component or object as taught by DeStefano, in order to obtain a system that is able to element based help information. One would have been motivated to make such a combination because of the need to have an improved manner of integrating together the presentation of interface controls with instructional material as taught by DeStefano.

With respect to **dependent claim 10,** Little teaches the [*method where the displaying to the user of an indication of each program element type includes displaying name information associated with a program element type.*] (Little figure 1-28) Little shows the displaying of program types (objects and/or classes and methods) with naming information.

With respect to **dependent claim 11,** claim 11 incorporates substantially similar subject matter as claimed in claim 9 and is rejected along the same rationale.

With respect to **dependent claim 12,** Little teaches the [*method where program element types are stored in schemas defining valid computer programs at different levels of abstraction.*] (Little figure 1-28) Little shows program trees in class diagrams that depict different levels of abstraction.

With respect to **dependent claim 13,** Little teaches the [*method wherein program element types are stored as definitions within the computer program.*] (Little figure 1-28) Little shows that classes and methods are stored as methods or classes within the computer program.

With respect to **dependent claim 14,** Little teaches the [*method where a program element type that is a specific instance of a more general program element type is defined by an is a relationship.*] (Little figure 1-28) Little shows objects that inherit from a parent class, which are "is a" relationships.

In regard to **Independent claim 22,** Little teaches a computer system for providing help information for a computer program, comprising:

- *A data structure storing a hierarchy of schemas defining computer programs, each schema specifying program element types of a computer program at different levels of abstraction*; (Little figure 1-28) Little shows program trees in class diagrams that depict data as it is stored in a database.
- *A store within a computer program having program elements, each program element having a program element type*; (Little figure 20 and page 10, column 2, lines 1-30) Little teaches a RDMS database for storing the class diagrams that represent the program elements.
- *A component that identifies a derivation of program element types for a program element of the computer program from the stored hierarchy of schemas*; (Little figure 1-28) Little shows the class diagram and the tree node (component) that shows the inherited class from a parent class.

- *A component that displays an indication of program element types in the identified derivation; and a component that displays information associated with a program element type selected from the displayed indication of program element types.* (Little figure 1-28) Little shows the class diagram and the tree node (component) that shows the inherited class from a parent class.

Little fails to disclose:

- Providing help information for a program

DeStefano discloses a user interface control for displaying multiple levels of user information based on the amount of detail that the user needs to perform the task. Individual elements within the program, which can be component, parameter, method or other variables and attributes, are assigned a level identifier, which categorizes the information into levels of abstraction. DeStefano teaches an example combo box where the instructional material is provided on how to enter data into the combo box. The user is able to drive down further to get more detailed information on the range of input and then drive down another level to get the reasons for the ranges. (DeStefano column 8, lines 39-67 and column 9, lines 35-45). DeStefano and Little are analogous art because they are from the same field of endeavor of displaying user information about the graphical user interface.

Accordingly, It would have been obvious to one of ordinary skill in the art, having the teachings of Little and DeStefano before him at the time of the invention was made, to modify the system of Little to incorporate the multiple levels of help that can be assigned to any component or object as taught by DeStefano, in order to obtain a system that is able to element based help information. One would have been motivated to make such a combination because of the need to have an improved manner of integrating together the presentation of interface controls with instructional material as taught by DeStefano.

With respect to **dependent claim 23,** Little teaches [*the computer system where the identifying of a derivation is performed in response to a user selecting a program element of the computer program.*] (Little figure 1-28) Little shows the identifying of the inherited classes on the class diagram and if the user selects the item on the drawing the item is depicted in the tree node component on the left of drawings 17-20.

With respect to **dependent claim 24,** Little teaches [*the computer system where the displaying to the user of an indication of program element types includes displaying name information associated with a program element type.*] (Little figure 1-28) Little shows the displaying of program types (objects and/or classes and methods) with naming information.

With respect to **dependent claim 25,** Little teaches [*the computer system where the information associated with a program element type is stored as an attribute of the program element type.*] (Little figure 1-28) Little shows attributes within the class, which is of a specific type and would be stored or embedded within the class. An attribute is descriptive information within a class.

With respect to **dependent claim 26,** Little teaches [*the computer system where a program element type is stored as a definition within the computer program.*] (Little figure 1-28) Little shows a class and the elements of a program and the classes are stored as a class within the program and have keywords identifying them as classes.

With respect to **dependent claim 27,** Little teaches [*the computer system where a program element type that is a specific instance of a more abstract program element type is specified by an is a relationship.*] (Little Figure 1-28) Little shows objects that inherit from a parent class, which are "is a" relationships.

In regard to **Independent claim 28,** Little teaches a computer-readable medium containing instructions for controlling a computer system to provide information for a computer program, by a method comprising:

- *Providing a computer program having program elements, each program element having a program element type, the program element types being defined by a hierarchy of schemas specifying program element types at different levels of abstraction*; (Little figure 1-28) Little shows a computer program with classes, methods, and attributes that are of a specific type and of different levels of abstraction. Little also teaches the files are executed on a server or within a workstation (Little page 16, column 2, lines 1-30).
- *Identifying a derivation of program element types for a program element of the provided computer program from the hierarchy of schemas*; (Little figure 1-28) Little shows the class diagram and the tree node (component) that shows the inherited class from a parent class.
- *Displaying an indication of program element types in the identified derivation; and* (Little figure 1-28) Little teaches which classes inherit or implement or are abstract classes by the common notation shown in the UML drawings.
- *Displaying information associated with a program element type selected from the displayed indication of program element types.* (Little figure 1-28) Little teaches that when a user selects the class or method the tree node will highlight the user selection and open the element on the right which will display information about the element.

With respect to **dependent claim 29,** Little teaches [*the computer-readable medium where the identifying of a derivation is performed in response to a user selecting a program element of the computer program.*] (Little page 16, column 2, lines 1-30 and figures 1-28) (Little figure 1-28) Little shows the class diagram and the tree node (component) that shows the inherited class from a parent class. Little shows that when the user selects the element the interface will highlight the item and show the base object it has inherited from. Little also teaches the execution of the files within the program on a server.

With respect to **dependent claim 30,** Little teaches [*the computer-readable medium where the information associated with a program element type is stored as an attribute of the program element type.*] (Little figure 1-28) Little shows attributes within the class, which is of a specific type and would be stored or embedded within the class. An attribute is information within a class.

With respect to **dependent claim 31,** Little teaches [the computer-readable medium where a program element type is stored as a definition within the computer program.] (Little figure 1-28) Little shows a class and the elements of a program and the classes are stored as a class within the program and have keywords identifying them as classes.

With respect to **dependent claim 32** Little teaches [The computer-readable medium where a program element type that is a specific instance of a more abstract program element type is specified by an is a relationship.] (Little figure 1-28) Little shows objects that inherit from a parent class, which are "is a" relationships.

In regard to **Independent claim 33,** Little teaches a method for providing help information for a computer program, the method comprising:

- *Providing a program tree representation of the computer program, the program tree having program elements and specifying a programmatic relationship for program elements of the program tree*; (Little figure 20 and page 10, column 2, lines 1-30) Little teaches a class diagram that represents a program and that contains nodes or icons graphics that are configured within a tree. Each node is an object or class with the respective inherited or dependent classes. Little shows the tree visually on the right and on the left in the non-visual tree component. The node type is the class type as specified in the diagram.
- *Identifying a derivation for a program element based on the specified programmatic relationship; and displaying information associated with the identified derivation* (Little figure 20). Little teaches the display of the hierarchical class structure in a class diagram and displays the class diagram

Little fails to expressly disclose:

- Providing help information for a computer program

DeStefano discloses a user interface control for displaying multiple levels of user information based on the amount of detail that the user needs to perform the task. Individual elements within the program, which can be component, parameter, method or other variables and attributes, are assigned a level identifier, which categorizes the information into levels of abstraction. DeStefano teaches an example combo box where the instructional material is provided on how to enter data into the combo box. The user is able to drive down further to get more detailed information on the range of input and then drive down another level to get the reasons for the ranges. (DeStefano column 8, lines 39-67 and column 9, lines 35-45). DeStefano and Little are analogous art because they are from the same field of endeavor of displaying user information about the graphical user interface.

Accordingly, It would have been obvious to one of ordinary skill in the art, having the teachings of Little and DeStefano before him at the time of the invention was made, to modify the system of Little to incorporate the multiple levels of help that can be assigned to any component or object as taught by DeStefano, in order to obtain a system that is able to element based help information. One would have been motivated to make such a combination because of the need to have an improved manner of integrating together the presentation of interface controls with instructional material as taught by DeStefano.

With respect to **dependent claim 34,** Little teaches the [*method where the programmatic relationship is based on program trees representing the computer program with different levels of abstraction.*] (Little figure 1-28) Little shows program trees in class diagrams that depict different levels of abstraction

With respect to **dependent claim 35,** Little teaches the [*method where the programmatic relationship is based on the hierarchy of operators and operands in the program tree.*] (Little figure 1-28) Little shows the inherited classes are organized in a hierarchy, as they would be used for program operation.

With respect to **dependent claim 36,** Little teaches [*the method where the programmatic relationship is based on the organization of the computer program.*] (Little figure 1-28) Little shows the inherited classes are organized in a hierarchy, as they would be used for program operation.

With respect to **dependent claim 37** Little teaches [*the method where the derivation includes a list of program elements representing the programmatic relationship.*] (Little figure 1-28) Little shows the classes and methods organized within relationships of inheritance and composition and on how the methods communicate with one another.

With respect to **dependent claim 38,** Little teaches the [method wherein the displayed information includes an indications of the programmatic relationships.] ((Little figure 1-28) Little shows the indications in the tree node of the nature of the relationship of an item within a class, a method and a package.

With respect to **dependent claim 39, 40 and 41,** as indicated in the above discussion, Little teaches/discloses every element of claim 38.

Little fails to expressly disclose/teach the [method where a user selects an indication of a programmatic relationship, displaying help information associated with the selected programmatic relationship and related program elements]

DeStefano discloses a user interface control for displaying multiple levels of user information based on the amount of detail that the user needs to perform the task. Individual elements within the program, which can be component, parameter, method or other variables and attributes, are assigned a level identifier, which categorizes the information into levels of abstraction. DeStefano and Little are analogous art because they are from the same field of endeavor of displaying user information about the graphical user interface.

Accordingly, It would have been obvious to one of ordinary skill in the art, having the teachings of Little and DeStefano before him at the time of the invention was made, to modify the system of Little to incorporate the multiple levels of help that can be assigned to any component or object as taught by DeStefano, in order to obtain a system that is able to element based help information. One would have been motivated to make such a combination because of the need to have an improved manner of integrating together the presentation of interface controls with instructional material as taught by DeStefano.

In regard to **Independent claim 42,** Little teaches a method for providing help information for a computer program, the method comprising:

- *Providing a program tree representation of the computer program, the program tree having program elements; providing a hierarchical programmatic relationship for program elements of the program tree*; (Little Figures 1-28) Little teaches a program tree representation in a class diagram which shows program elements and the hierarchical relationships within the trees.
- *Receiving from a user a selection of a program element of the program tree; identifying a derivation of the provided hierarchical programmatic relationship for the identified program element; and* (Little figures 1-28) Little shows that when a user selects a given class, method or attribute the item is highlighted and the tree node expands giving an indication of the relationships within the program and a diagram appears in the window on the right.

Little fails to expressly disclose:

- Displaying to the user the identified derivation; receiving from user a selection of programmatic relationship of the displayed derivation; **retrieving help information associated with the selected programmatic relationship**; and displaying to the user the **retrieved help information**.

DeStefano discloses a user interface control for displaying multiple levels of user information based on the amount of detail that the user needs to perform the task. Individual elements within the program, which can be component, parameter, method or other variables and attributes, are assigned a level identifier, which categorizes the information into levels of abstraction. DeStefano teaches an example combo box where the instructional material is provided on how to enter data into the combo box. The user is able to drive down further to get more detailed information on the range of input and then drive down another level to get the reasons for the ranges. (DeStefano column 8, lines 39-67 and column 9, lines 35-45). DeStefano and Little are analogous art because they are from the same field of endeavor of displaying user information about the graphical user interface.

Accordingly, It would have been obvious to one of ordinary skill in the art, having the teachings of Little and DeStefano before him at the time of the invention was made, to modify the system of Little to incorporate the multiple levels of help that can be assigned to any component or object as taught by DeStefano, in order to obtain a system that is able to element based help information. One would have been motivated to make such a combination because of the need to have an improved manner of integrating together the presentation of interface controls with instructional material as taught by DeStefano.

With respect to **dependent claim 43**, Little teaches the [method where the programmatic relationship is based on program trees representing the computer program with different levels of abstraction.] (Little figure 1-28) Little shows program trees in class diagrams that depict different levels of abstraction

With respect to **dependent claim 44** Little teaches the [*method where the programmatic relationship is based on the hierarchy of operators and operands in the program tree.*] (Little figures 1-28) Little shows a hierarchy of operators and operands within the class diagrams.

With respect to **dependent claim 45**, Little teaches the [*method where the programmatic relationship is based on the organization of the computer program.*] (Little figures 1-28) Little shows the relationships of the objects are expressed to depict the operation of the program.

With respect to **dependent claim 46**, as indicated in the above discussion, Little teaches/discloses every element of claim 42.

Little fails to expressly teach/disclose the [*method including identifying program elements related to the selected program element wherein the displaying includes displaying help information associated with the related program elements.*]

DeStefano discloses a user interface control for displaying multiple levels of user information based on the amount of detail that the user needs to perform the task. Individual elements within the program, which can be component, parameter, method or other variables and attributes, are assigned a level identifier, which categorizes the information into levels of abstraction. DeStefano and Little are analogous art because they are from the same field of endeavor of displaying user information about the graphical user interface.

Accordingly, it would have been obvious to one of ordinary skill in the art, having the teachings of Little and DeStefano before him at the time of the invention was made, to modify the system of Little to incorporate the multiple levels of help that can be assigned to any component or object as taught by DeStefano, in order to obtain a system that is able to element based help information. One would have been motivated to make such a combination because of the need to have an improved manner of integrating together the presentation of interface controls with instructional material as taught by DeStefano.

References to specific columns, figures or lines should not be limiting in any way. The entire reference provides disclosure related to the claimed invention.

Conclusion

7. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

U.S. Patent No. 6,502,233 B1 to Vaidyanathan et al issued Dec. 31, 2002 and filed Nov. 13, 1998, and discloses an automated help system for reference information.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Steven B. Theriault whose telephone number is (571) 272-5867. The examiner can normally be reached on M-F 7:00 - 3:30 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Heather Herndon can be reached on (571) 272-4136. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

SBT

BA HUYNH
PRIMARY EXAMINER